Preprint
UCRL-JC-141022

Profiling Users in the UNIX OS Environment

V. N. P. Dao, R. Vemuri, S. J. Templeton

This article was submitted to
International Computer Science Conventions, Wollongong,
Australia, December 12-15, 2000

September 29, 2000

U.S. Department of Energy

Lawrence
Livermore
National
Laboratory

This work was performed under the auspices of the United States Department of Energy by the University of California, Lawrence Livermore National Laboratory under contract No. W-7405-Eng-48.

Profiling Users in the UNIX OS Environment

Vu N. P. Dao [1]
daol@llnl.gov

Rao Vemuri [1,2]
rvemuri@ucdavis.edu

Steven J. Templeton [2]
templets@cs.ucdavis.edu

[1] Lawrence Livermore National Laboratory, 7000 East Ave., Livermore, CA 94551
[2] University of California, Davis, One Shields Ave., Davis, CA 95616

Abstract

This paper presents results obtained by using a method of profiling a user based on the login host, the login time, the command set, and the command set execution time of the profiled user. It is assumed that the user is logging onto a UNIX host on a computer network.

The paper concentrates on two areas: short-term and long-term profiling. In short-term profiling the focus is on profiling the user at a given session where user characteristics do not change much. In long-term profiling, the duration of observation is over a much longer period of time. The latter is more challenging because of a phenomenon called concept or profile drift. Profile drift occurs when a user logs onto a host for an extended period of time (over several sessions).

I. Introduction

Profiling is a technique of grouping individuals or things into groups or categories based on certain features such as appearance, traits, situations, etc. The term profiling probably strikes a negative feeling in many people because most are aware of negative applications of profiling in news headlines. Nevertheless there are many benefits and useful applications in user profiling.

Following are a few examples of profiling. Constructive examples of profiling include grocers buying vegetable or fruit products based on color and firmness of the produce, while destructive examples include management not promoting employees based on color, race and gender. Other examples of profiling include law enforcement officers stopping certain types of people for questioning after a crime has occurred. In a nutshell, profiling is a classification procedure that groups pertinent information of an event or situation together so that people can make better decisions pertaining to that event or situation. In many ways, the results obtained from profiling proved accurate, although one can debate the legal and ethical issues involved in many profiling applications.

The science of profiling has been successfully used in many important application areas – most notably in law enforcement. A recent high-profile case was that of identifying the 'Unabomber'. After going over a manifesto purportedly authored by the 'Unabomber' [1], the FBI came up with a profile. Most of the profile characteristics proved to be correct when the 'Unabomber' was apprehended. Other examples include identifying the author of a piece of litertary work based on that author's usage of words, grammar, and so on.

Profiling (or, equivalently classifying) has many applications in the realm of modern computer and information technology. By profiling users one can have a better understanding of the users' computer usage patterns. The results can then be used to allocate system resources more efficiently and to provide better services within a networked (or collaborative) environment. In other areas, an ability to infer user preferences from user behavior patterns has many applications in Internet-based commerce. An ability to infer user intentions from user behavior has applications in detecting and arresting computer-based crimes. The long-term goal of our work is to use user profiling as one of the ingredients in detecting intruders into a secure networked environment.

This paper is organized as follows. A brief review of the literature on profiling computer users and the direction of this paper is given in section II. Section III discusses the system resources that can be used for profiling. To set the stage for the experimental results presented in the other sections, this section discusses the topology of the computer network, and describes how the data were collected. Section IV discusses the essential parameters used to profile the users. Section V covers short term user profiling, or more specifically user profile within a few sessions on an individual host. Section VI covers long term user profiling, or user profiling over many sessions of a host. This section further dwells into the drift in user profiling known in the literature as profile or concept drift. Section VII provides a summary of the work presented in this paper. Section VIII concludes the paper. This section listed the results found and discusses future work in user profiling especially in the area of computer security. Section IX listed the references used in this paper, and Appendix A listed the results.

II. Literatures Reviewed and the Direction of this Paper on Profiling Computer Users

The area of profiling computer users for detecting intrusions was first mentioned in Denning's paper [2] on building a model for intrusion detection in 1987. From that time many others elaborated to include different ways of profiling the users. Some of these included those of Obaidat and Sadoun [3], whose work concentrated on identifying computer users through the keystroke dynamics. Lane and Brodley [4] concentrated their works in monitoring the UNIX commands that the user typed. They introduced "concept drift" [5] to take into account changes in the user profile. Warrender, Forrest, and Pearlmutter [6] used system calls into the kernel of an operating system to profile user usages for intrusion detection. Profiling computer users for applications in computer security has thrived in recent years.

Aside from the research works mentioned above, many others worked in the practical aspects of intrusion detection also pointed out the need for an accurate user profile model. Among these were Bace [7], and Northcutt [8]; both authors talked about ways of detecting intruders logging into their networks through the use of user profiling. From their experiences, both authors classified computer break-ins into two main categories – inside and outside intruders. Inside intruders have authorized use of the computer network; whereas outside intruders do not have authorization. These two authors concluded that many applications in computer science, especially computer security (i.e. intrusion detection) can benefit from computer user profile.

The direction of this paper is to build a foundation on user profiling for future works in intrusion detection. To build a reliable intrusion detection system, Bass [9] suggested that multisensors should be used. Multisensing is a method of combining data from multiple and diverse sensors and sources in order to make inferences about events, activities, and situation. Thinking along this line, this paper presents a method of profiling a user through multiple parameters from the process accounting log of the system. The multiple parameters user profile obtained here will be used as one of the many components for our subsequent work in intrusion detection.

III. System Resources for User Profiling

Before any work in user profiling is done, it is important to focus on the essential data based on the system resources that one has, and the system available to users. This section begins with a description of the computer network topology in service. Next the section discussed how the user data was monitored and logged.

(a) Computer Network Topology

The topology of our computer network at the University of California, Davis, Computer Security Laboratory consists of a number of computer hosts from a variety of manufacturers. All hosts run one of four operating systems: Sun Solaris, Sun OS, Free BSD, and LINUX. Both the Sun Solaris and Sun OS run on the workstations, while the Free BSD and LINUX run on the PCs. There are a few Macintosh personal computers connected to the lab network but we limited our work to the UNIX workstations and PCs only.

Since all the workstations are connected together, most run the same software applications. Nevertheless, a few workstations were dedicated to run specially licensed software applications to save cost on network licensing. On the other hand, the PCs have separately installed software applications running on them. The PCs share one commonality with the workstations – the UNIX operating system.

Any user with a valid account can log onto any workstation or host within the computer network through two means. The first login means is to log in from any physical host in the lab. The second method of accessing the lab network is through dial up modems. Once connected to any host within the lab, the user has the capability to remotely log on to other computers within the laboratory network.

(b) Data Monitoring and Logging

In the UNIX operating system, there is a process accounting program that is running in the background [10,11] of the operating system. The process accounting program keeps tab of the computer resources. Some of these computer resources are keyboard input, time of keyboard usage, CPU usage, memory usage, cache memory usage, buffer usage, etc.

By invoking the process accounting program to run [12] and piping the output to a logfile, we were able to collect the inputs each user typed on the keyboard.

IV. Profiling Users Through Their Most Important Characteristics

One of the key difficulties in user profiling in the context of networked environment is that there are many different kinds of users and these can be categorized using a vast number of variables. Some of the variables, such as gender, physical and intellectual capabilities, and communication skills do not change at all or change slowly. Some variables such as stress, fatigue, computer-related experience, skills at typing at the keyboard, preferences for certain types of information, propensity to use certain commands, may show some drift with time and experience. User profiles can also be developed on the basis of interaction features a user prefers (menu-based interaction, command line interaction, via function keys, etc). Researchers have also used machine learning techniques to track user actions and construct models for user preference [13].

Each user is a unique individual with a unique set of characteristics. When faced with the same problem or situation, each individual has a unique perspective of solving or looking at that situation. The hypothesis is that these individual behavioral characteristics can be extracted from the log files of each user.

Our approach depends on learning characteristic sequences of actions generated by users. The underlying hypothesis is that a user responds in a similar manner to similar situations, leading to repeated sequences of actions. Indeed, the existence of command alias mechanisms in many UNIX command interpreters supports the idea that users tend to perform many repeated sets of actions, and that these sequences differ on a per-user basis. It is the differences in characteristic sequences that we attempt to use to profile users in order to differentiate users.

With the above ideas in mind, we chose four parameters in the captured logfile to profile the users. Although these four parameters are not an exhaustive list of parameters in the logfile, we found empirically that they contain more than the adequate information we need to profile the user on. These four user profile features are the login host, the login time, the command set, and the command execution time. We will go in detail on each of these features below:

(a) Login Host

In our network, as well as most UNIX operating system networks, the users have the freedom to connect to any host on the network. Many times, the users want to keep separate applications running on separate hosts because of software license agreements or have a preferred host that they want to log on to (e.g. the user's personal workstation or assigned host). On different occasions, users might want to keep the work related to certain applications or projects specific to a host on the network. Thus, it is important to keep track of the host that a user is logged on because the same user can have a different user characteristic from host to host.

(b) Login Time

Most users tend to have a preferred time window to do their work. For instance, a nocturnal person whose normal computer login time is between 10:00PM to 2:00AM is unlikely to log onto the network at 8:00AM in the morning except in some unexpected situations. Likewise, a user whose regular schedule is from 10:00 AM to 6:00PM rarely logs in at night, say from 12:00AM to 6:00AM. Thus the login time is considered to be a useful profiling parameter.

(c) Command Sets

The command set is perhaps the most important parameter to profile a user on. It is a more distinguished trait that makes a user unique. However the UNIX command set available to users is large. To profile a user based on all commands in the UNIX command set is an overwhelming task. Yet, almost all UNIX users utilize a portion of the available command set.

To simplify our task, we went over all the logfiles and selected a set of 100 UNIX commands that all users most frequently used. The result, in the order of the frequency of their usage, is listed in Table 1 (i. e. the command used most frequently is listed first). Table 1 is used as a secondary command set to profile a user.

The primary command set to profile a user is his own. In determining a user's command set, the following rule applies. If the number of commands in the user's repertoire of commands is greater than 100, then only the most used 100 commands in his log file are used. However, if the user command set is N, (i.e., N < 100) then all of the N commands will be used. In addition (100 - N) top commands in Table 1 will be used to augment the user's command set. The conditions below re-state the logic discussed.

Given: CS = Commands in the user's Command Set

N = The number of commands actually used

```
if (CS > 100)
    CS = 100 most used commands
else if (CS < 100)
    CS = N + [Top (100-N) commands*]
End
```

*commands from Table 1 that are not in CS.

1 sh
2 stty
3 sed
4 mail
5 [
6 dtfile
7 frm
8 in.telne
9 gen.pl
10 groff
11 date
12 sendmail
13 hostname
14 uudemon.
15 tty
16 tetex.cr
17 perl
18 emacs
19 dot
20 grotty
21 row
22 tcsh
23 cat
24 su
25 test
26 more
27 utmp_upd
28 whoami
29 top
30 troff
31 column
32 grids
33 updatedb
34 find
35 ps
36 frcode
37 lpNet
38 mkdir
39 w
40 file
41 dtexec
42 chmod
43 logger
44 bash
45 pt_chmod
46 logrotat
47 xterm
48 rdate
49 gzip
50 xhost
51 dtscreen
52 rm
53 vi
54 less
55 grep
56 tmpwatch
57 rlogin
58 tr
59 ln
60 top-sun4
61 msgchk
62 gabriel_
63 in.rshd
64 sort
65 amd
66 finger
67 telnet
68 tput
69 resize
70 gtbl
71 rsh
72 mv
73 id
74 clear
75 crond
76 uuxqt
77 quota
78 pwd
79 domainna
80 mesg
81 uname
82 ptbl
83 cp
84 id.pl
85 run-part
86 uusched
87 ping
88 df
89 xlock
90 lpr
91 awk
92 ls
93 login
94 chown
95 atrun
96 man
97 movemail
98 gunzip
99 last
100 in.rlogi

Table 1: 100 Frequently Used Command Set by all users

(d) Command Execution Time

The final parameter that was monitored for user profiling is the execution time of each command. The command execution time parameter tracks how much time a command is required to run after a user hits the return key. In UNIX, any user can modify a command or creating an alias command to do a series of tasks. For instance a directory listing is typically defined as 'ls', however the same command can be used to list files in the current directory with different options, as in 'ls – la'. 'ls – la' would do a long listing of all files in the current directory with a complete listing of when the files were created, and their size, etc. Furthermore, any experienced UNIX user can use the same command to do other tasks such as deleting that directory, (i.e. define 'ls' to do 'rm' of the directory or the hard drive). The latter one is known as a trojan command – i.e. the command is defined for doing a specific task [14] other than intended. Most trojan commands are malicious in nature.

To prevent unexpected execution time of commands outside of their scopes, the tracking of the execution of these commands would isolate those commands that took more CPU cycle time to run than normal.

V. Host User Profile

Using the logfiles referred in section III, we then proceeded on profiling the user according to the four features – the login host, the login time, the command set, and the command execution time.

As mentioned in subsection IVa, some hosts on our computer network have different applications running on them. We decided to profile the users on each host individually. In the short-term profiling case, several steps were involved. The first step was to parse the data into each host that the profiled user had logged on. Here we selected the command set according to the rules presented in the command set subsection (subsection IVc). In the second step, we divided the commands into a one-week period and counted the number of occurrence of each command in that one-week interval. In the third step, we determined when the users logged onto the lab network by the login time. Finally we looked at the command execution time to see if any alias (i.e. trojan) commands were run.

From the above four steps, we generated a command set and a login time for each user. For illustration purposes, we presented eight different plots from two different users that we profiled on in Appendix A. The command execution time profile did not vary much (i.e. no trojan commands were detected), thus was not plotted. However, it is necessary to keep track of the time each command was run.

Figures 1a and 1b illustrate the command set and login time of User 1 on Host A, while figures 1c and 1d illustrate the same User 1 on Host B. Figures 2 (a & b) and figures 2 (c & d) represent the User 2 on Host C and D respectively. In figures 1 (a & c), 2 (a & c) the X-axis indicates the command set of the profiled user; the Y-axis indicates the profiled user's time login in a one-week increments. The Z-axis indicates the percentage of occurrence of each command. Similarly, in figures 1b, 1d, 2b, and 2d the X-axis indicates the time of log in (starting from 12:00AM and ending at 11:00PM). The Y-axis represents the weekly increment in time, and the Z-axis represents the percentage of usage in a one-hour duration when the user is logged in.

Figures 1a and 1c are the command plots of the same user (User 1) on host A and B. From observation, one can see that these two plots do not exhibit any similar pattern. Likewise figures 1b and 1d are the time plot of the same user on host A and B do not show any similar correlation. After inspecting the plots for User 1 on other hosts, we can see some similarity of the command and or time plots to either figures 1a and 1b or figures 1c and 1d.

In a similar comparison, the profile of User 2 (figures 2a and 2b, and 2c and 2d) exhibits some similarity in the same command set and or time usage pattern across the two hosts (Hosts A & B).

The results of Users (1 & 2) above are but two cases out of the 28 users that we profiled in our network. Of all of the users we had profiled in the short-term case, we learned that a user profile is a function of the host. The same user can have different or similar profiles on different hosts. This can be attributed to the fact that a user used a computer host for a specific application or need. In other words, if the same user is using different applications on different hosts, then his profiles on these hosts will be the same. However if the same user is using different applications on different hosts, then his profiles on these hosts will be different.

VI. Host User Profile with Concept or Profile Drift

When a person works in the same environment for an extended period of time, it is highly likely that he will adapt and change his style to fit his environment. This is because the user becomes more familiar with the system, or that he has discovered a better way of doing things [5]. The same can be said of doing one's work. The nature of the job might change over time. As a result, a user modifies his command set to fit his new situation. These changes in user's behavior correspond to the long-term profiling. The change in work habit, work environment or application in one's account is known in the literature as concept or profile drift.

Two situations contribute to profile drift – natural profile drift and forced profile drift. Natural profile drift occurs when a user slowly adapts to his environment through learning or experiences. This happens when the user learned a new command or a new way of doing the same thing. For instance, after working on a project for a while, most workers will become more experienced and perform their job better. Natural profile drift is gradual and constantly occurring in the user profile. On the other hand, forced profile drift occurs when the user abruptly changes his profile. This usually happens when the user changes one or more of the following: work environment, responsibility or job. The user is forced to change his normal working profile in order to accommodate his new role. Usually when forced profile drift occurs, the user will subsequently undergo a natural progression drift once he is comfortable in his new environment.

Profile drift is evident in the figures 1 (a,b,c,d) and 2 (a,b,c,d). In the natural profile drift, the command figures are more linear while the time figures stay constant. In the second case of forced profile drift, both the command and time figures have abrupt changes. After that more gradual changes would occur in the command and time plots.

VII. Summary

We used the built-in process accounting log of the UNIX operating systems on the hosts to log the users' usage. From the users' logfiles we determined the four most essential parameters to be used in profiling. The four profile parameters were the login host, the login time, the command set, and the command execution time. The login host and time correspond to the identity of the host and the time that the profiled user logs onto the network. The command set is the 100 most frequently used commands that the profiled user uses. If the profiled user uses less than 100 commands, then we appended his command set with the most frequently used commands from Table 1. We found no noticeable difference in the user's command execution time and decided to include this feature in our future work in intrusion detection.

The login host, login time and command set were adequate in profiling users in both short term and long term profiling sessions. Moreover, in short term profiling, we found that the user profile is dependent on the host that he logs on. In long term profiling there also exists profile drift. Thus in the long term profiling case a user profile appears to depend upon both the host and login time.

VIII. Conclusion and Future Work

This paper has demonstrated that the host, the login time, and the UNIX command set can be used to profile a user with a high degree of accuracy. Two important points were learned. First, the user profile is host dependent. The same user could have different profiles on different hosts. This is due to the fact that user profiling is a function of the applications residing on the host. The second point was that some profile drift occurred over time. Profile drift occurs in two ways – natural profile drift and forced profile drift. Both are due to the fact that users will change their profile to fit their environment.

Further work in this area can be a monitoring of different system parameters such as memory usage, page fault usage, buffer over, etc. Perhaps an entirely new process accounting system to track the desired parameters for user profiling is also possible. Also the results obtained in this paper were in the form of figures and observations. The conclusions we made were by observing those figures in the appendix. It is important to come up with a quantitative measurement of these results. This can be accomplished if the users' profiles are used in actual applications such as intrusion detection.

Furthermore, as we have been making the connection of the work presented here to that of intrusion detection throughout this paper, it is important to point in that direction for future research. The work in this paper represents one foundation in a multisensing system to be used in detecting intruders logging onto a computer network.

IX. References

[1] Unabomber, "Industrial Society and Its Future", *The Washington Post*, Sept. 19, 1995.

[2] D. Denning, "An Intrusion Detection Model", *IEEE Transactions on Software Engineering*, 1987.

[3] M. Obaidat, B. Sadoun, "Verification of Computer Users Using Keystroke Dynamics," *IEEE Trans. On Systems, Man and Cybernetics*, Vol. 27, No. 2, pp. 261-269, Apr. 1997.

[4] T. Lane, C. Brodley, "An Application of Machine Learning to Anomaly Detection", *http://mow.ecn.purdue.edu/~terran/facts/research/research.html*, 1997.

[5] T. Lane, C. Brodley, "Temporal Sequence Learning and Data Reduction for Anomaly Detection", *http://mow.ecn.purdue.edu/~terran/facts/research/research.html*, 1998.

[6] C. Warrender, S. Forrest, B. Pearlmutter, "Detecting Intrusions Using System Calls: Alternative Data Models", *IEEE,* Nov. 1999.

[7] R. Bace, *Intrusion Detection*, Macmillan Technical Publishing, 2000.

[8] S. Northcutt, *Network Intrusion Detection —An Analyst's Handbook,* New Riders Publishing, 1999.

[9] T. Bass, "Intrusion Detection Systems and Multisensor Data Fusion", *Communications of the ACM*, Vol. 43, No. 4, Apr. 2000.

[10] S. Garfinkel, G. Spafford, *Practical UNIX & Internet Security*, O'Reilly, 1996.

[11] S. Coffin, *UNIX System V Release IV: The Complete Reference*, McGraw Hill, 1990.

[12] E. Nemeth, G. Snyder, S. Seebass, T. Hein, *UNIX System Administration Handbook*, 2nd ed. Prentice Hall PTR, 1995

[13] P. Maes, "Agents that Reduce Work and Information Overload", *Communications of the ACM*, Vol. 37, July 94.

[14] L. Klander, *Hacker Proof – The Ultimate Guide To Network Security*, Jamsa Press, 1997.

Appendix A – Results

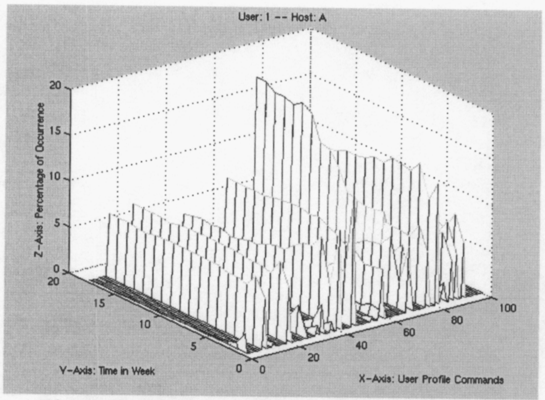

Figure 1a: Command Plot of User 1 – Host A
X axis – Command Set
Y axis – Weekly Increment in Time
Z axis – % of Command Usage

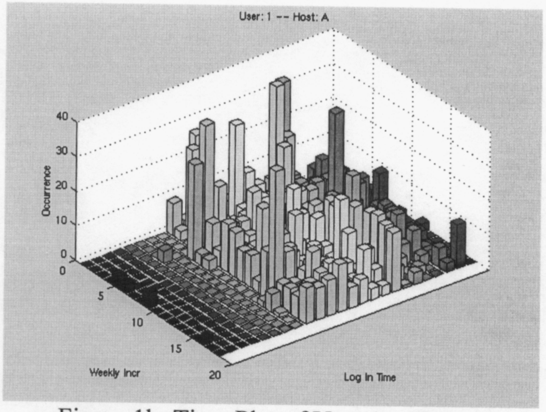

Figure 1b: Time Plot of User 1 – Host A
X axis – Time Login [12:00AM - 11:00PM]
Y axis – Weekly Increment in Time
Z axis – % of Command Usage

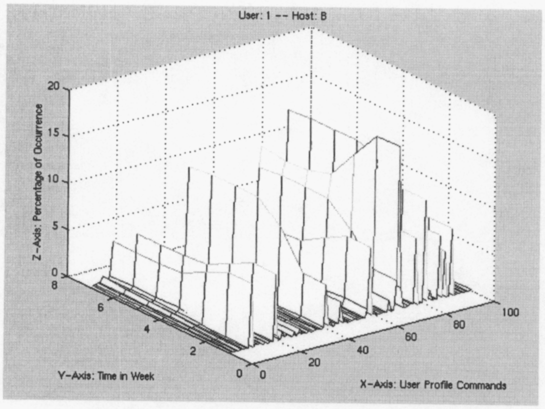

Figure 1c: Command Plot of user 1 – Host B
X axis – Command Set
Y axis – Weekly Increment in Time
Z axis – % of Command Usage

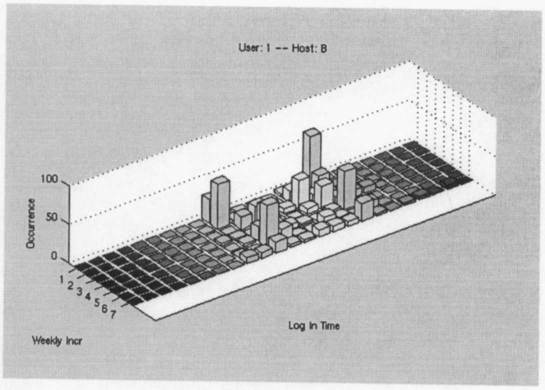

Figure 1d: Time Plot of User 1 – Host B
X axis – Time Login [12:00AM - 11:00PM]
Y axis – Weekly Increment in Time
Z axis – % of Command Usage

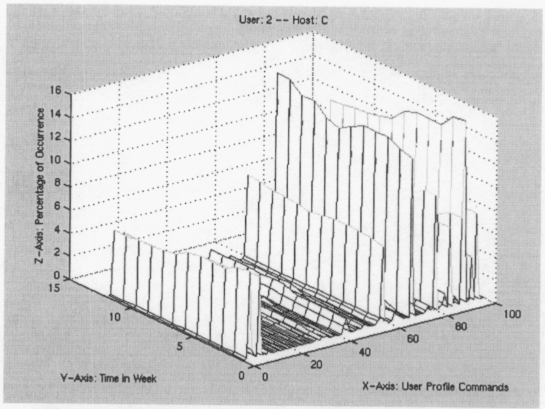

Figure 2a: Command Plot of User 2 – Host C
X axis – Command Set
Y axis – Weekly Increment in Time
Z axis – % of Command Usage

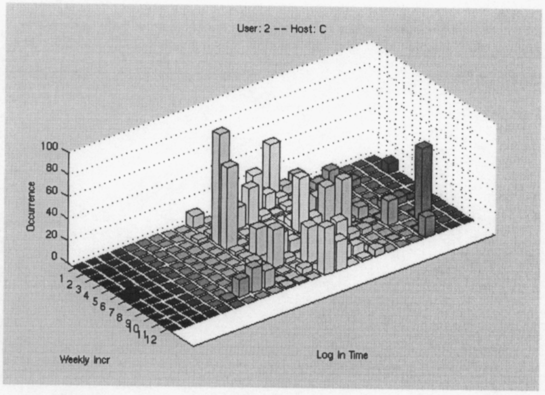

Figure 2b: Time Plot of User 2 – Host C
X axis – Time Login [12:00AM - 11:00PM]
Y axis – Weekly Increment in Time
Z axis – % of Command Usage

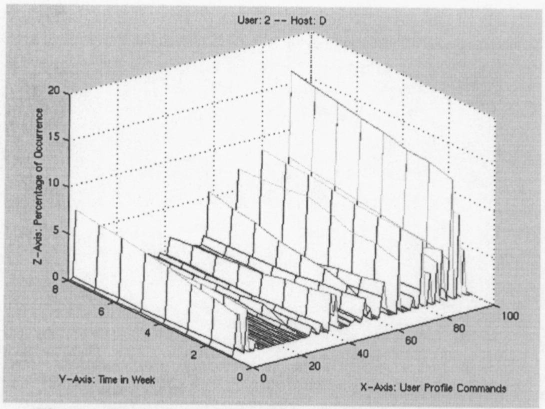

Figure 2c: Command Plot of User 2 – Host D
X axis – Command Set
Y axis – Weekly Increment in Time
Z axis – % of Command Usage

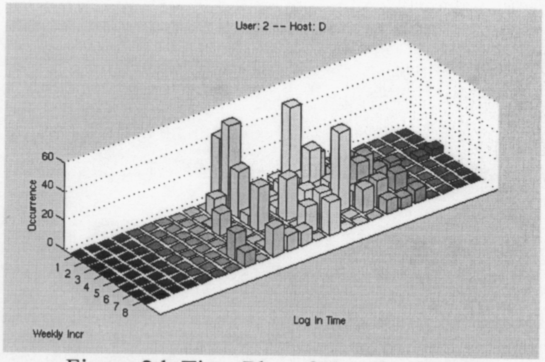

Figure 2d: Time Plot of User 2 – Host D
X axis – Time Login [12:00AM - 11:00PM]
Y axis – Weekly Increment in Time
Z axis – % of Command Usage